



Fitness Merkezi Yazılımı

Yazılım Mühendisliği Ana Bilim Dalı

Dönem Projesi

Abdurrahman POLAT

Proje Danışmanı: Prof. Dr. Doğan AYDIN

Ocak 2024

Öz

Fitness Merkezleri için hazırladığım bu projede, Fitness merkezi üyelerinin kişisel bilgi , vücut gelişimi, gelir kayıtları , hizmet bilgilerinin kayıt edilerek ve satınalma muhasebe gibi modüllerle gelir gider takibinin daha iyi düzgün bir şekilde takip edilip istendiği zaman çeşitli grafiklerle raporlanıp üye kişiye çıktı olarak yada email olarak gönderilmesi amaçlanmaktadır. Program Python yazılım diliyle (Qt) PySide6 framework kullanılarak kullanıcı dostu karmaşık olmayan bir arayüze sahiptir. Veri tabanı olarak Sqlite kullanılmıştır.

Anahtar Sözcükler: Fitness Software,Fitness,Fitness Program,Qt,PySide6,Fitness Rapor

Abstract

This project, prepared for Fitness Centers, is used when it is desired to keep track of income and expense records better, with modules such as personal information of Fitness center people, body development, income, registration of service information and purchase accounting, and the member can be reported with various reports as a printout or e-mail. It is intended as a postal service in the region. The program is coded in an easy-to-use, uncomplicated way using the PySide6 framework in the Python programming language (Qt). Sqlite was used as the database.

Key Words: Fitness Software, Fitness, Fitness Program, Qt, PySide6, Fitness Report

Teşekkür

Proje çalışmasına katkılarından dolayı, projenin gelişimine önemli katkılarda bulunan arkadaşlarıma ve danışmanlarıma teşekkür ederim.

İçindekiler

Öz	1
Abstract	2
Teşekkür	3
1. Giriş	8
2. Genel Bilgiler	9
2.1 Ekipmanlar ve Aletler:	9
2.2 Antrenman Alanları:	10
2.3 Grup Dersleri:	10
2.4 Kişisel Antrenman:	11
2.5 Sauna ve Spa Hizmetleri:	11
2.6 Üyelik Modelleri:	11
2.7 Eğitim Programları:	12
3. İşleyiş	13
3.1 Üye kaydı açma.....	13
3.2 Üye bilgileri görüntüleme ve güncelleme	14
3.2.1 Kişi kartı	14
3.2.2 Çalışma Programı	15
3.2.3 Gelişim Ekle	17
3.2.4 Hizmet Ekle	23
3.2.5 Kişiyeye özel programları listele.....	29
3.2.6 Şablon oluşturulması	30
3.2.7 Pilates ve benzeri işlemler	33
3.2.8 Muhasebe gelir gider tablosu.....	37
3.2.9 Hizmetlerin oluşturulması.	42
3.3 Satınalma , fatura ve stok yönetimi.....	44
3.4 Firma (Cari kart tanımlama)	50

3.12 Kantin ve ürün satışı	57
Kaynaklar	60

Şekiller

2.1 Ekipman ve Aletler.....	9
2.2 Antrenman Alanları.....	10
2.3 Grup dersleri.....	10
3.4 Kişisel Antrenman.....	11
2.5 Sauna.....	11
2.7 Çalışma planı.....	12
3.1 Üye kayıt ekranı.....	14
3.2.1 Kişi kartı.....	15
3.2.2 Kişiyeye özel çalışma planı.....	16
3.2.3.1 Gelişim verileri.....	22
3.2.3.2 Gelişim verileri baskı ön izleme.....	23
3.2.4.1 Hizmet listesi.....	27
3.2.4.2 Eklenen hizmetler.....	28
3.2.5 Üyelere ait çalışma programları.....	29
3.2.6 Şablon tanımlama ve düzenleme.....	32
3.2.7 Pilates tablosu.....	37
3.2.8.1 Muhasebe gelir.....	41
3.2.8.2 Muhasebe gider.....	41
3.2.8.3 Muhasebe istatistik.....	42
3.2.9.1 Hizmetler listesi.....	43
3.2.9.2 Yeni Hizmet ekle.....	43
3.3.1 Satınalma stok listesi ve satınalma geçmişi.....	47
3.3.2 Ürün tanımlama listesi.....	47
3.3.3 İşlenen faturalar.....	48
3.3.4 Fatura işleme.....	48

3.3.5 İşlenen faturanın detaylarını girme.....	49
3.4.1 Firma/Cari kartlar.....	52
3.4.2 Firma /Cari kart ekleme.....	52
3.4.3 Firma / Cari görüntüleme-güncelleme ve detaylar.....	54
3.5 Kantin Modülü.....	59

1. Giriş

Günümüzde sporla uğraşan kişi sayısı hızla artmaktadır. Fitness merkezlerine olan talepler bu yerlerin ihtiyacı olan kayıt tutma kişiye özel programlar oluşturma gibi durumları ortaya çıkarmaktadır. Bu noktada kayıtların daha düzgün daha kısa sürede ulaşılabilir olması bu tür bilgilerin kısa sürede doğru bir şekilde raporlanması için yazılıma ihtiyaç duyulmaktadır.

Fitness merkezi yazılımı ile bu ihtiyaçların giderilmesi hedeflenerek uygulama ile kişilerin gelişimi takip edilip, satın alma süreçlerini yönetmek, ürün satışlarını gerçekleştirip bu ürünlerle ilgili satış raporlarını sunmak gelir gider muhasebesini yapmak amaçlanmaktadır.

2. Genel Bilgiler

Fitness merkezi, genellikle spor ve egzersiz faaliyetlerini destekleyen, fitness ekipmanları, antrenman alanları, grup dersleri ve dięer fitness hizmetlerini sunan tesislerdir. Bu tür tesisler, bireylerin fiziksel aktivite düzeyini artırmalarına, saęlıklarını sürdürmelerine veya geliřtirmelerine yardımcı olmak amacıyla kurulmuřlardır. Fitness merkezleri, genellikle řu özellikleri içerebilir:

2.1 Ekipmanlar ve Aletler:

Cardio makineleri, serbest aęırlık ekipmanları, direnç makineleri, egzersiz topu gibi çeřitli fitness ekipmanlarını içerir.



řekil 2.1 Aęırlık ve Bench Press sehpası

2.2 Antrenman Alanları: Bireylerin kendi antrenmanlarını yapabilecekleri geniş alanlar, ağırlık kaldırma alanları, matlar ve esneme alanları içerir.



Şekil 2.2 Antrenman alanı

2.3 Grup Dersleri: Yoga, pilates, spinning, aerobik gibi çeşitli grup dersleri sunar. Bu dersler genellikle eğitmen eşliğinde gerçekleştirilir.



Şekil 2.3 Pilates Grup dersleri

2.4 Kişisel Antrenman: Bireylere özel antrenman programları ve kişisel eğitmen hizmetleri sunar.



Şekil 2.4 Kişisel Antrenman

2.5 Sauna ve Spa Hizmetleri: Bazı fitness merkezleri, sauna, buhar odası veya spa hizmetleri gibi ekstra olanakları da içerebilir.



Şekil 2.5 Sauna

2.6 Üyelik Modelleri: Aylık veya yıllık üyelik sistemleri üzerinden hizmet sunabilirler.

2.7 Eğitim Programları: Fitness merkezleri, başlangıç seviyesinden ileri seviyeye kadar farklı fitness programları sunabilir.

PROGRAM DETAYLARI								
HAFTA GÜNÜ	ALET / MAKİNA	AÇIKLAMA	BÖLGE	AĞIRLIK	SET	TEKRAR	ARA	
1	Koşu bandı	15 dk genel ısınma	tüm vücut					
2	PAZARTESİ	dumbell shoulder press	sehpada dik oturur şekilde	omuz	10 kğ	5	8	30 sn
3		biceps curl machine	makinede ön kol çalışma	kol	10 kğ	6	8	45 sn
4		bench press machine	makine ile üst göğüs	göğüs	15 kğ	4	7	1 dk
5	ÇARŞAMBA	Koşu bandı	15 dk genel ısınma	tüm vücut				
6		Tricep pushdown	makine ile ayakta	arka kol	10 kğ	7	10	30 sn
7		Rope Pushdown	makine ile ayakta tam geriye çekilecek	arka kol	10 kğ	4	7	45 sn
8	Leg Press machine	makine ile bacak çalışma (%80 itme)	bacak	30 kğ	6	10	30 sn	
9	CUMA	Koşu bandı	15 dk genel ısınma	tüm vücut				
10		dumbell	ayakta kalf kasları çalış	kalf	20 kğ	4	12	30 sn
11		Leg Extention machine	makinede üst bacak (tam kaldır)	bacak	30 kğ	5	8	1 dk
12		Sehpa	sehpa üzerinde Hyperextension	sırt		7	8	40 sn

Şekil 2.7 Çalışma planı

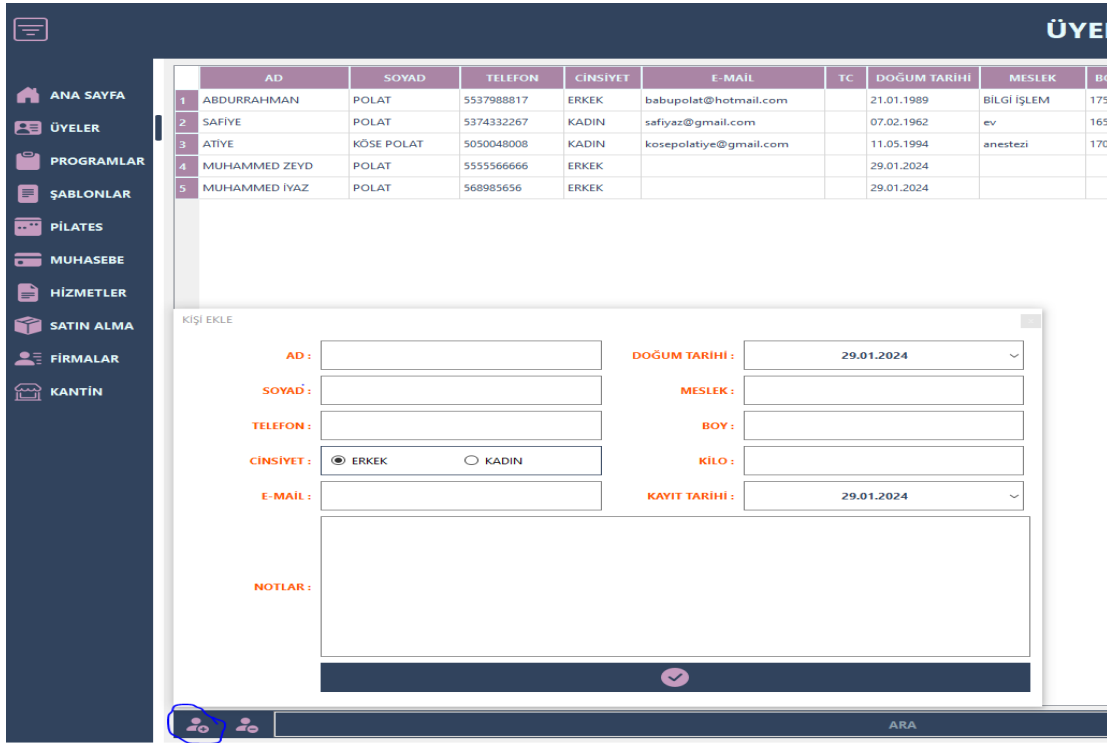
Bu tür tesisler, genellikle bireylerin spor yapma alışkanlıklarını destekleyerek sağlıklı bir yaşam tarzına ulaşmalarına yardımcı olmayı amaçlar. Fitness merkezleri, genellikle profesyonel eğitimci ve personel tarafından yönetilir ve üyelerine sağlıklı yaşam konusunda rehberlik yaparlar.

3. İşleyiş

3.1 Üye kaydı açma

Programda üyenin bilgileri ve buna bağlı oluşturulacak olan kişisel gelişimi, ödeme geçmişi, eğitim programlarını, gelişim grafikleri gibi bilgilerin kaydedilmesi için gerekli olan bölümdür. Her üye ye ait benzersiz bir kimlik oluşturulur.

```
def sql_kisiEkle(self, ad, soyad, telefon, cinsiyet, email, tc, dogumtarihi, meslek, boy, kilo,
kayittarihi,
notlar, foto):
    self.sql_baglan()
    query = (
        "insert into kisikartlari (ad, soyad, telefon, cinsiyet, email, tc, dogumtarihi, meslek, boy,
kilo, "
        "kayittarihi,notlar,foto) values(?,?,?,?,?,?,?,?,?,?,?,?,?)")
    param = (
        ad, soyad, telefon, cinsiyet, email, tc, dogumtarihi, meslek, boy, kilo, kayittarihi, notlar,
        foto)
    try:
        self.cur.execute(query, param)
        self.con.commit()
        self.sql_kapat()
        return "Yeni Kişi Oluşturuldu"
    except sqlite3.IntegrityError:
        e = "Aynı Telefon No sistemde kayıtlı"
        self.con.rollback()
        self.sql_kapat()
        return e
```



Şekil 3.1 Üye kayıt ekranı

3.2 Üye bilgileri görüntüleme ve güncelleme

Üye kaydının güncellenmesi program, vücut ölçümleri ve hizmet ekleme çıkarma güncelleme gibi işlemlerin yapılması için gereklidir ilgili üyeye çift tıklanarak kişi detayları sayfasına yönlendirilir. Yönlendirme esnasında tablodaki kişi bilgilerini bir sonraki açılacak olan sayfaya aktarılır. Bu işlem aşağıdaki kod parçasıyla gerçekleşir.

3.2.1 Kişi kartı

Üyenin detaylarının görüntülenmesi için ilgili tablodan üyeye çift tıklanır

def kisidetaylari(self):

 kisidetay = FitnesKisiDetay(self.screenize)

 kisibilgileri = lambda x:

self.tableWidget_kayitlar.item(self.tableWidget_kayitlar.currentRow(), x)

 idd = kisibilgileri(0).text()

 ad = kisibilgileri(1).text()

 soyad = kisibilgileri(2).text()

 telefon = kisibilgileri(3).text()

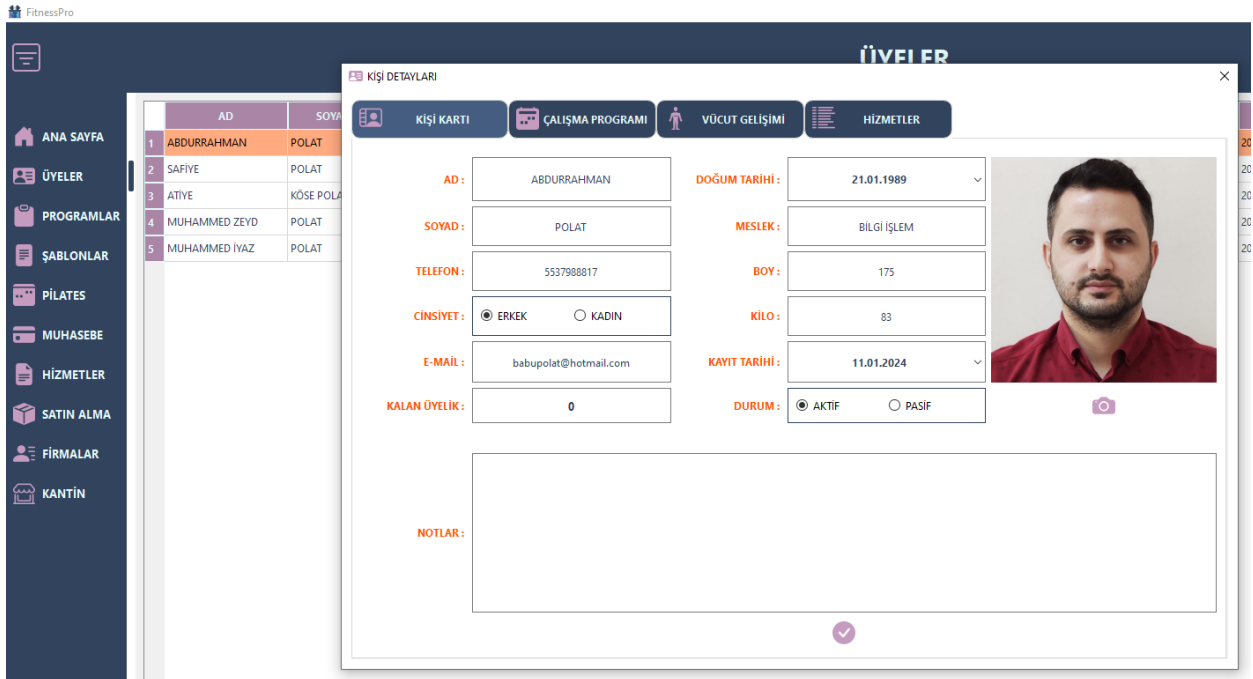
 cinsiyet = kisibilgileri(4).text()

 email = kisibilgileri(5).text()

```

tc = kisibilgileri(6).text()
dogumtarihi = kisibilgileri(7).text()
meslek = kisibilgileri(8).text()
boy = kisibilgileri(9).text()
kilo = kisibilgileri(10).text()
kayittarihi = kisibilgileri(11).text()
aktifpasif = kisibilgileri(12).text()
notlar = kisibilgileri(13).text()
kisidetay.kisibilgileridoldur(idd, ad, soyad, telefon, cinsiyet, email,
tc, meslek, kayittarihi, dogumtarihi, boy, kilo, aktifpasif, notlar)
kisidetay.exec()
kisidetay.destroy(True, True)
del (idd, ad, soyad, telefon, cinsiyet, email, tc, meslek, kayittarihi, dogumtarihi, boy, kilo,
kisibilgileri)
kisidetay.deleteLater()
self.kisitablosu()

```



Şekil 3.2.1 Kişi Kartı

3.2.2 Çalışma Programı

Kişiyeye özel çalışma programlarının tanımlandığı alandır. Bu kısımda öncelikle sol üst kısımda bulunan buton ile başlangıç ve bitiş aralığı tanımlanan bir program oluşturulur. Daha sonra

aşağıda bulunan + butonu veya daha önceden şablon olarak tanımlamış olduğumuz çalışma programını ilgili sayfadaki şablonlar butonuna tıklayarak hızlı bir şekilde ekleyebiliriz.

```
class FitnessCalismaEkle(QDialog, Ui_CalismaEkle):
    def __init__(self, kisi_id):
        super().__init__()
        self.setupUi(self)
        self.kisi_id = kisi_id
        self.setWindowFlag(Qt.WindowType.Sheet)
        self.setWindowFlag(Qt.WindowType.WindowStaysOnTopHint)
        self.dateEdit_baslangic.setDate(QDate.currentDate())
        self.dateEdit_bitis.setDate(QDate.currentDate())
        self.pushButton_kaydet.clicked.connect(self.yeniprogramolustur)

    def yeniprogramolustur(self):
        sql = SqlFitnessKisiCalismaProgrami()
        sql.sql_calismaEkle(self.kisi_id, self.lineEdit_aciklama.text(),
self.dateEdit_baslangic.date().toPython(),
        self.dateEdit_bitis.date().toPython())
        self.close()
        del sql
```

BAŞLANGIÇ	BİTİŞ	GÜN	KALAN
24.01.2024	24.02.2024	31 GÜN	26 GÜN

HAFTA GÜNÜ	ALET / MAKİNE	AÇIKLAMA	BÖLGE	AĞIRLIK	SET	TEKRAR
1	Koşu bandı	15 dk genel ısınma	tüm vücut			
2	dumbell shoulder press	sehpada dik oturur şekilde	omuz	10 kg	5	8
3	biceps curl machine	makinede ön kol çalışma	kol	10 kg	6	8
4	bench press machine	makine ile üst göğüs	göğüs	15 kg	4	7
5	Koşu bandı	15 dk genel ısınma	tüm vücut			
6	Tricep pushdown	makine ile ayakta	arka kol	10 kg	7	10
7	Rope Pushdown	makine ile ayakta tam geriye çekilecek	arka kol	10 kg	4	7
8	Leg Press machine	makine ile bacak çalışma (%80 itme)	bacak	30 kg	6	10

Şekil 3.2.2 Kişiyeye özel çalışma programı

3.2.3 Gelişim Ekle

Üyelerin gelişimlerini takip etmek çalışılan programın ne derecede etkili olduğunu gözlemlemek ve buna bağlı olarak programda değişiklik, güncelleme yada ekleme yapmak daha verimli bir gelişime katkı sağlayacaktır. Burda üyeye ait belirli periyodlarla alınan vücut ölçülerinin kayıtlarını tutmak ve bunları grafiksel olarak gösterebilmek önemlidir. Bu verilerin görselleştirilmesi için Bar ve Line serileri kullanıldı ayrıca ölçümlerde ilk ve son ölçüm arasındaki farkı analiz eden bir görsel kullanıldı. (<https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QLineSeries.html>) (<https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QPieSeries.html>)

```
def karsilastirmaverileri(self):
```

```
    colum = [(self.tableWidget_vucutgelisimi.isColumnHidden(x), x) for x in
              range(self.tableWidget_vucutgelisimi.columnCount()) if x not in (0, 1, 13)]
    columno = []
    for tf, colno in colum:
        if tf != True:
            columno.append(colno)
    datas = {self.tableWidget_vucutgelisimi.horizontalHeaderItem(x).text(): [] for x in
             columno if
              x not in (0, 1, 13)}

    veri = lambda row, x: self.tableWidget_vucutgelisimi.item(row, x).text()

    for no, x in zip(columno, datas.keys()):
        for row in range(self.tableWidget_vucutgelisimi.rowCount()):
            datas[x].append(((veri(row, 13), veri(row, no))))

    return datas

def karsilastirmaekle(self):
    if self.tableWidget_vucutgelisimi.rowCount() == 0:
        return False
```

```

self.chartviewline.chart.removeAllSeries()
self.chartviewbar.chart.removeAllSeries()
tarih = [x for x in self.karsilastirmaverileri().values()]
ilktarih = tarih[0][0][0]
sontarih = tarih[0][-1][0]
self.label_ilkkarsilastirma_tarih.setText(ilktarih)
self.label_sonkarsilastirma_tarih.setText(sontarih)

self.chartviewline.x_axis.setMin(QDateTime().fromString(self.label_ilkkarsilastirma_tarih.t
ext(), "yyyy-MM-dd"))

self.chartviewline.x_axis.setMax(QDateTime().fromString(self.label_sonkarsilastirma_tarih.t
ext(), "yyyy-MM-dd"))
self.chartviewline.veriekle(self.karsilastirmaverileri())
self.chartviewbar.veriekle(
    [QDate.fromString(self.tableWidget_vucutgelisimi.item(x, 13).text(), "yyyy-MM-
dd").toString("MM-yyyy") for x
    in range(self.tableWidget_vucutgelisimi.rowCount())], self.karsilastirmaverileri())
self.verikarsilastir(self.karsilastirmaverileri())

def verikarsilastir(self, veriler):
    veri = []
    line = []
    name = [x for x in veriler]
    for i in veriler.values():
        veri.append(i[0][1])
        veri.append(i[-1][1])
        veri.append(int(i[-1][1]) - int(i[0][1]))

    for i in self.frame_gelisimverileri.findChildren(QFrame):
        if i.accessibleName() in name:
            for c in i.children():
                if type(c) == QLineEdit:
                    line.append(c)

```

```

for lin, value in zip(line, veri):
    lin.setText(str(value))
self.veridurumu()

def veridurumu(self):
    up = QPixmap(u":/icon/RindeIcons/chart_arrow_up.png")
    down = QPixmap(u":/icon/RindeIcons/chart arrow_down.png")
    same = QPixmap(u":/icon/RindeIcons/activitycolored.svg")
    for i in self.frame_gelisimverileri.children():

        for c in i.children():
            if c.objectName().endswith("_fark"):
                if len(c.text()) > 0:
                    i.show()
                    s = i.findChildren(QLabel)
                    for lab in s:
                        if lab.objectName().endswith("_durum"):
                            if int(c.text()) > 0:
                                lab.setPixmap(up)
                            elif int(c.text()) < 0:
                                lab.setPixmap(down)
                            else:
                                lab.setPixmap(same)
                        else:
                            i.hide()

class FitnessLineChartView(QChartView):
    def __init__(self):
        super(FitnessLineChartView, self).__init__()
        self.font = QFont()
        self.font.setFamily("Roboto")
        self.font.setPointSize(8)
        self.setRenderHint(QPainter.RenderHint.Antialiasing)
        self.chart = QChart()
        self.chart.setMargins(QMargins(0, 0, 0, 0))

```

```

self.setChart(self.chart)
self.x_axis = QDateTimeAxis()
self.x_axis.setLabelsFont(self.font)
self.x_axis.setLabelsAngle(-20)
self.x_axis.setFormat("yyyy-MM-dd")

self.y_axis = QValueAxis()
self.y_axis.setLabelsFont(self.font)
self.y_axis.setMin(0)
self.y_axis.setMax(250)
self.y_axis.setLabelFormat("%i cm/kg")

```

```
def veriekle(self, veriler):
```

```

    self.chart.addAxis(self.x_axis, Qt.AlignmentFlag.AlignBottom)
    self.chart.addAxis(self.y_axis, Qt.AlignmentFlag.AlignLeft)
    for col, row in veriler.items():
        series = QLineSeries()
        series.setPointsVisible(True)
        series.setName(str(col))
        for v in row:
            date = QDateTime().fromString(v[0], "yyyy-MM-dd")
            series.append(date.toMsecsSinceEpoch(), int(v[1]))
        self.chart.addSeries(series)
        series.attachAxis(self.x_axis)
        series.attachAxis(self.y_axis)

```

```
class FitnessBarChartView(QChartView):
```

```

    def __init__(self):
        super(FitnessBarChartView, self).__init__()
        self.chart = QChart()
        self.chart.setMargins(QMargins(0, 0, 0, 0))
        self.setChart(self.chart)
        self.setRenderHint(QPainter.RenderHint.Antialiasing)

```

```

self.chart.legend().setVisible(True)
self.chart.legend().setAlignment(Qt.AlignmentFlag.AlignTop)

self.font = QFont()
self.font.setFamily("Roboto")
self.font.setPointSize(8)

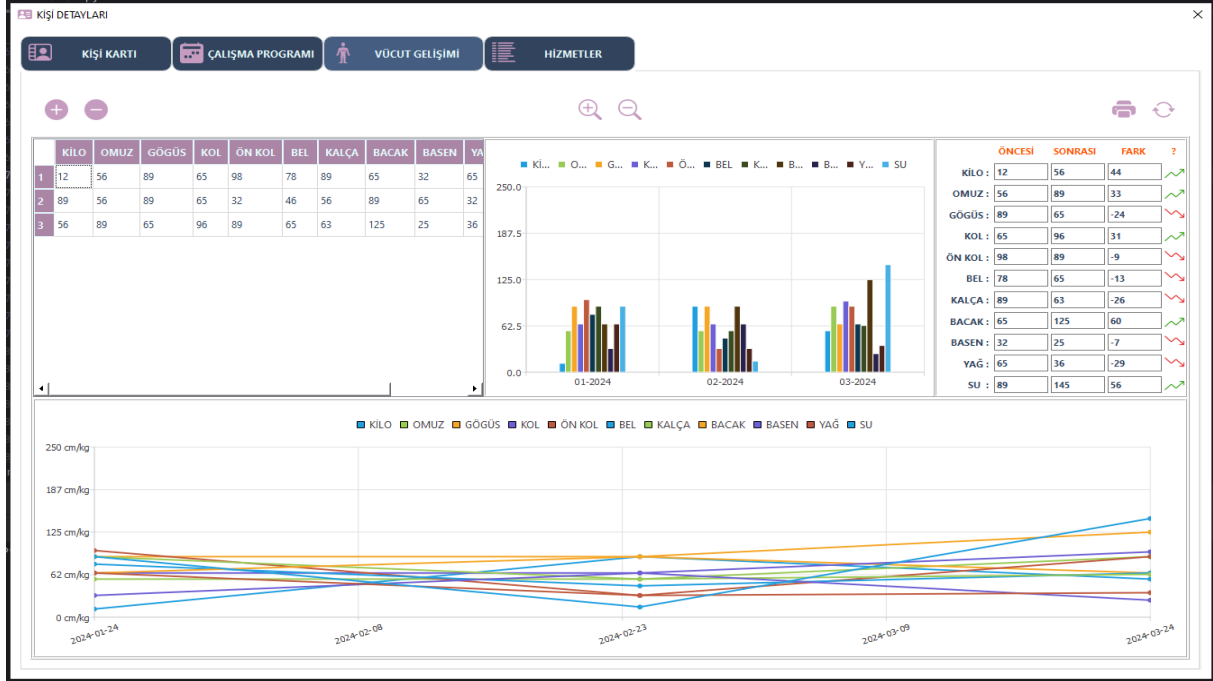
self.axis_x = QBarCategoryAxis()

self.axis_y = QValueAxis()
self.axis_y.setLabelsFont(self.font)
self.axis_y.setRange(0, 250)

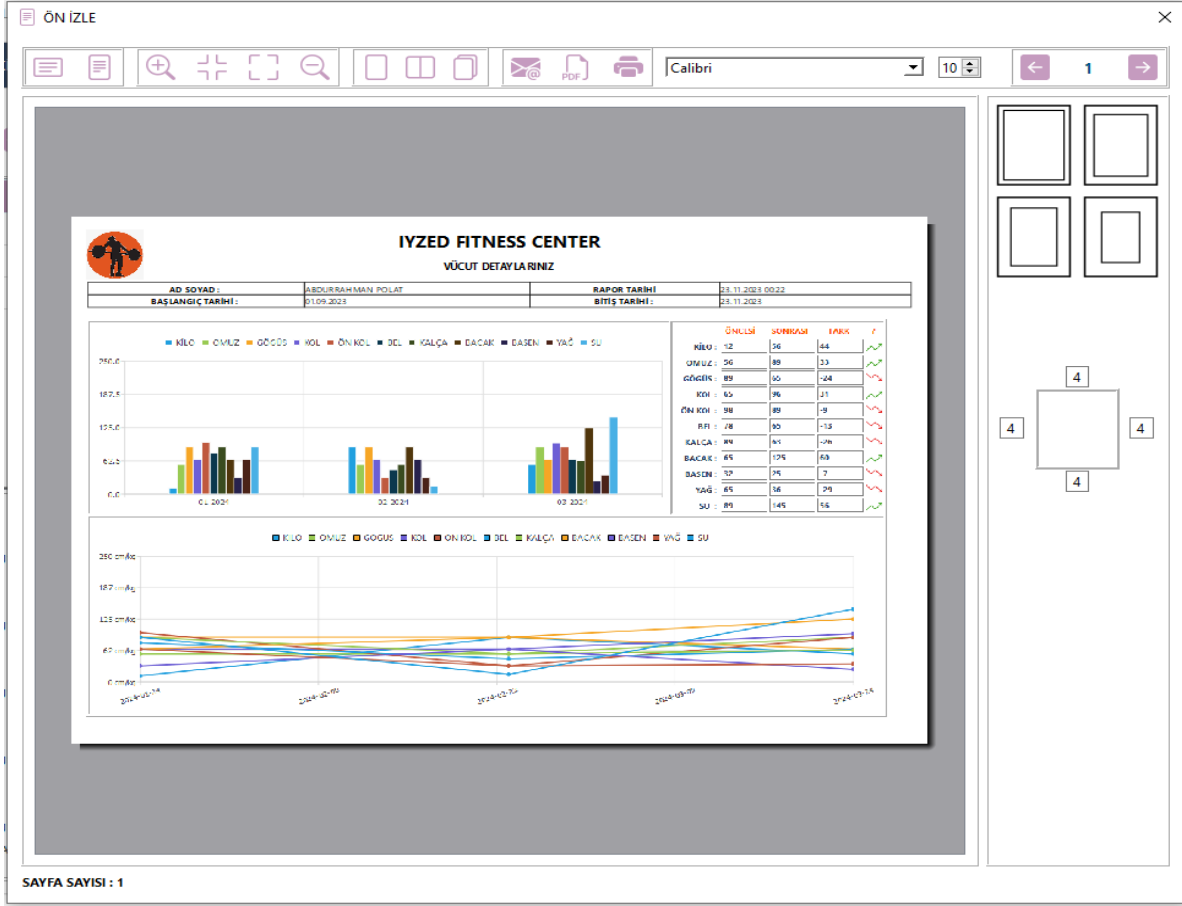
self.chart.addAxis(self.axis_y, Qt.AlignmentFlag.AlignLeft)
self.chart.addAxis(self.axis_x, Qt.AlignmentFlag.AlignBottom)

def veriekle(self, kategori, datas):
    series = QBarSeries()
    self.axis_x.clear()
    self.axis_x.append(kategori)
    self.axis_x.setLabelsFont(self.font)
    for i in datas.items():
        bar = QBarSet(i[0])
        bar.setLabelFont(self.font)
        bar.append([int(c[1]) for c in i[1]])
        series.append(bar)
    self.chart.addSeries(series)
    series.attachAxis(self.axis_x)
    series.attachAxis(self.axis_y)

```



Şekil 3.2.3.1 Gelişim verileri



Şekil 3.2.3.2 Gelişim verileri baskı ön izleme.

3.2.4 Hizmet Ekle

Üyenin satın aldığı hizmetleri görüntülemek için kullanılan alandır. Bu kısımda öncelikle hizmet butonuna tıklanıp hizmet oluşturulur daha sonra hizmet ekle butonundan açılan sayfada oluşturulan hizmete sistemde tanımlanmış ve üyeye eklemek istediğimiz hizmetler seçilip kaydedilir.

def hizmetekle(self):

hizmetekle = FitnessHizmetEkle(self.lineEdit_id.text())

hizmetekle.exec()

hizmetekle.destroy(True, True)

hizmetekle.deleteLater()

self.hizmettablo()


```

class FitnessHizmetEkle(QDialog, Ui_HizmetEkle):
    def __init__(self, idd):
        super().__init__()
        self.setupUi(self)
        self.restoreColumnState()
        self.id_kisi = idd
        self.setWindowFlag(Qt.WindowType.Sheet)
        self.setWindowFlag(Qt.WindowType.WindowStaysOnTopHint)
        self.sqlHizmetler = SqlFitnessHizmetler()
        self.sqlHizmet = SqlFitnessHizmetEklenen()
        self.sqlHizmetDetay = SqlFitnessHizmetEklenenDetay()
        self.sqlMuhasebe = SqlFitnessMuhasebeGelir()
        self.tableWidget_hizmettablosu.itemDoubleClicked.connect(self.hizmetekle)
        self.tableWidget_eklenenhizmetler.itemDoubleClicked.connect(self.hizmetasil)
        self.lineEdit_iskonto.editingFinished.connect(self.iskontoyap)
        self.lineEdit_geneltoplam.textChanged.connect(self.iskontokontrol)
        self.pushButton_kaydet.clicked.connect(self.hizmetkaydet)
        self.setvalidator()
        self.hizmet = []
        self.hizmetlistesi()
    def hizmetlistesi(self):
        tabloyuolustur(self.tableWidget_hizmettablosu, self.sqlHizmetler.sql_hizmetler())

    def iskontokontrol(self):
        if int(self.lineEdit_geneltoplam.text()) < 0:
            self.lineEdit_iskonto.setText("0")
            self.lineEdit_geneltoplam.setText("0")

    def iskontoyap(self):
        self.lineEdit_geneltoplam.setText(str(int(self.lineEdit_toplam.text()) -
int(self.lineEdit_iskonto.text())))

    def setvalidator(self):
        self.validator = QRegularExpressionValidator()

```

```

self.validator.setRegularExpression("[1-9]*,[1-9]")
self.lines = [x for x in self.children() if x.objectName().startswith("line")]
for i in self.lines:
    i.setValidator(self.validator)

def hizmetekle(self):

    item =
[self.tableWidget_hizmettablosu.item(self.tableWidget_hizmettablosu.currentRow(), x).text()
for x in
    range(0, self.tableWidget_hizmettablosu.columnCount())]
if item not in self.hizmet:
    self.hizmet.append(item)

self.tableWidget_eklenenhizmetler.setRowCount(len(self.hizmet))
for rowno, col in enumerate(self.hizmet):
    for colno, data in enumerate(col):
        item = QTableWidgetItem()
        item.setText(str(data))
        self.tableWidget_eklenenhizmetler.setItem(rowno, colno, item)
total = sum([int(x[3]) for x in self.hizmet])
self.lineEdit_toplam.setText(str(total))
self.lineEdit_geneltoplam.setText(str(int(total) - int(self.lineEdit_iskonto.text())))
self.iskontokontrol()
def hizmetsil(self):
    item =
[self.tableWidget_eklenenhizmetler.item(self.tableWidget_eklenenhizmetler.currentRow(),
x).text() for x
    in
    range(0, self.tableWidget_eklenenhizmetler.columnCount())]
if item in self.hizmet:
    self.hizmet.pop(self.hizmet.index(item))
self.tableWidget_eklenenhizmetler.setRowCount(len(self.hizmet))
for rowno, col in enumerate(self.hizmet):

```

```

for colno, data in enumerate(col):
    item = QTableWidgetItem()
    item.setText(str(data))
    self.tableWidget_eklenenhizmetler.setItem(rowno, colno, item)
total = sum([int(x[3]) for x in self.hizmet])
self.lineEdit_toplam.setText(str(total))
self.lineEdit_geneltoplam.setText(str(int(total) - int(self.lineEdit_iskonto.text())))
self.iskontokontrol()
def hizmetkaydet(self):
    id_hizmeteklenen = self.sqlHizmet.sql_hizmetlereklenenekle(self.id_kisi)
    for i in self.hizmet:
        i.insert(0, id_hizmeteklenen)
        i.pop(2)
        i.pop(2)
    self.sqlHizmetDetay.sql_hizmetlereklenendetayekle(self.hizmet)
    self.sqlMuhasebe.sql_muhasabegelirekle(id_hizmeteklenen,
self.lineEdit_geneltoplam.text(),
                                self.lineEdit_iskonto.text(),
                                self.radiobuttonGroup.checkedButton().accessibleName())
    self.close()

```

HİZMET EKLE

HİZMETLER

	HİZMET ADI	AÇIKLAMA	FİYAT
1	FİTNESS	3 AY	4500
2	PİLATES	3 AY	6000

HİZMET ARA

EKLENEN HİZMETLER

HİZMET ADI	AÇIKLAMA	FİYAT
------------	----------	-------

ÖDEME TÜRÜ

NAKİT KREDİ KARTI EFT/HAVALE BORÇ

TOPLAM TUTAR : TL

İSKONTO : TL

GENEL TOPLAM : TL



Şekil 3.2.4.1 Hizmet Listesi

KİŞİ DETAYLARI

KİŞİ KARTI ÇALIŞMA PROGRAMI VÜCUT GELİŞİMİ HİZMETLER

TARİH

2024-01-11 14:40:34

2024-01-26 09:37:42

ALINAN HİZMETLER

AÇIKLAMA	FİYAT	ÖDEME TÜRÜ
1 PİLATES 3 AY	NAKİT	6000
2 FITNESS 3 AY	NAKİT	4500

TOPLAM TUTAR : 10500 TL ÖDENEN TUTAR : 10500 TL İSKONTO : 0 TL KALAN TUTAR : 0 TL

Şekil 3.2.4.2 Eklenen hizmetler

3.2.5 Kişiyeye özel programları listele

Bu modülde üyelere ait çalışma programlarını tarih aralığı veya ad,soyad,telefon numarası gibi parametrelerle sorgulayıp listelemek ve üyenin çalışma programında güncellemeler yapmak yada çalışma programının dökümanını almak gibi işlemler yapılabilmektedir.

```
def calismaDetaylari(self):  
    if self.tableWidget_programgecmisi.selectedItems():  
        tabloyuolustur(self.tableWidget_programdetaylar, self.sqlCalisma.sql_calismaDetaylari(  
self.tableWidget_programgecmisi.item(self.tableWidget_programgecmisi.currentRow(),  
0).text()))  
        set_row_span(self.tableWidget_programdetaylar)
```

```
def sql_calismaDetaylari(self, idd):  
    self.sql_baglan()  
    query = ("select * from calismaprogramlaridetay where id_program=?")  
    param = (idd,)  
    result = self.cur.execute(query, param).fetchall()  
    self.sql_kapat()  
    return result
```

ÜYE PROGRAMLARI				PROGRAM DETAYLARI								
	AÇIKLAMA	AD	SOYAD	HAFTA GÜNÜ	ALET / MAKİNA	AÇIKLAMA	BÖLGE	AĞIRLIK	SET	TEKRAR	ARA	
1	BAŞLANGIÇ SEVİYE	SAFIYE	POLAT	1		Koşu bandı	15 dk genel ısınma	tüm vücut				
2	ORTA SEVİYE 1 AYLIK FITNESS PROGRAMI	ABDURRAHMAN	POLAT	2		dumbbell shoulder press	sehpada dik oturur şekilde	omuz	10 kg	5	8	30 sn
3				PAZARTESİ		biceps curl machine	makinede ön kol çalışma	kol	10 kg	6	8	45 sn
4						bench press machine	makine ile üst göğüs	göğüs	15 kg	4	7	1 dk
5						Koşu bandı	15 dk genel ısınma	tüm vücut				
6				6		Tricep pushdown	makine ile ayakta	arka kol	10 kg	7	10	30 sn
7				ÇARŞAMBA		Rope Pushdown	makine ile ayakta tam geriye çekilecek	arka kol	10 kg	4	7	45 sn
8						Leg Press machine	makine ile bacak çalışma (%80 itme)	bacak	30 kg	6	10	30 sn
9						Koşu bandı	15 dk genel ısınma	tüm vücut				
10				10		dumbbell	ayakta kalf kasları çalış	kalf	20 kg	4	12	30 sn
11				CUMA		Leg Extension machine	makinede üst bacak (tam kaldır)	bacak	30 kg	5	8	1 dk
12						Sehpa	sehpa üzerinde Hyperextension	sırt		7	8	40 sn

Şekil 3.2.5 Üyelere ait çalışma programları

3.2.6 Şablon oluşturulması

Belirli özelliklere sahip kişilere sürekli aynı veya büyük oranda benzer özelliklere sahip çalışma programları hazırlamak zaman alabilir. Bu nedenle şablon modülü ile daha önceden belirli kategorilere ait çalışma programları tanımlanıp benzer vücut, beslenme, çalışma ve uyku düzenine veya benzeri özelliklere sahip kişilere tanımlı şablonlardan hızlı bir şekilde çalışma programı tanımlanabilir. Şablon sayfasında sol alt köşede bulunan şablon ekleme butonu ile şablon ekleyip daha sonra + butonu ile şablon detaylarını oluşturabilirsiniz.

```
def sablonsayfasi(self):
```

```
    if self.tableWidget_programgecmisi.selectedItems():
```

```
        sablon = FitnessSablon()
```

```
        sablon.setSablon()
```

```
        sablon.exec()
```

```
    if sablon.sablonid != None:
```

```
        colum = (1, 2, 3, 4, 5, 6, 7, 8, 9)
```

```
        data = self.sqlSablon.sql_sablondetay(sablon.sablonid)
```

```
        veri = []
```

```
        for i in data:
```

```
            veri.append([i[x] for x in colum])
```

```
        for i in veri:
```

```
            i[0] =
```

```
self.tableWidget_programgecmisi.item(self.tableWidget_programgecmisi.currentRow(),  
0).text()
```

```
        self.sqlCalisma.sql_calismaDetayEkleToplu(veri)
```

```
        self.calismaDetaylari()
```

```
        sablon.destroy(True, True)
```

```
        sablon.deleteLater()
```

```
def sablonlar(self):
```

```
    self.tableWidget_sablonlar.clearSelection()
```

```
    self.tableWidget_sablondetay.clearSelection()
```

```
    self.tableWidget_sablondetay.setRowCount(0)
```

```
    tabloyuolustur(self.tableWidget_sablonlar, self.sqlSablon.sql_sablonlar())
```

```

def sablonekle(self):
    self.sqlSablon.sql_sablonekle("YENİ ŞABLON")
    tabloyuolustur(self.tableWidget_sablonlar, self.sqlSablon.sql_sablonlar())

def sablonguncelle(self):
    self.sqlSablon.sql_sablonguncelle(
        self.tableWidget_sablonlar.item(self.tableWidget_sablonlar.currentRow(), 1).text(),
        self.tableWidget_sablonlar.item(self.tableWidget_sablonlar.currentRow(), 0).text())

def sablonsil(self):
    self.sqlSablon.sql_sablonsil(self.tableWidget_sablonlar.item(self.tableWidget_sablonlar.curre
ntRow(), 0).text())
    self.sablonlar()
    self.tableWidget_sablondetay.setRowCount(0)

def sablondetay(self):
    self.label_sablonadi.setText(
        str(self.tableWidget_sablonlar.item(self.tableWidget_sablonlar.currentRow(), 1).text()))
    tabloyuolustur(self.tableWidget_sablondetay, self.sqlSablon.sql_sablondetay(
        self.tableWidget_sablonlar.item(self.tableWidget_sablonlar.currentRow(), 0).text()))
    set_row_span(self.tableWidget_sablondetay)

def sablondetayekle(self):
    if self.tableWidget_sablonlar.selectedItems():
        self.pushButton_sablondetayguncelle.click()
        self.sqlSablon.sql_sablondetayekle(
            self.tableWidget_sablonlar.item(self.tableWidget_sablonlar.currentRow(), 0).text(),
            "", "", "", "", 0, 0, 0, 0)
        self.sablondetay()
    else:
        self.message.setText("Soldaki tablodan sablon seçiniz")
        self.message.exec()

```



```
def sablondetayguncelle(self):
```

```
    column = (2, 3, 4, 5, 6, 7, 8, 9, 0)
```

```
    datas = [[self.tableWidget_sablondetay.item(row, col).text() for col in  
              column] for row in
```

```
              range(0, self.tableWidget_sablondetay.rowCount())]
```

```
    self.sqlSablon.sql_sablondetayguncelle(datas)
```

```
def sablondetaysil(self):
```

```
    if self.tableWidget_sablondetay.selectedItems():
```

```
        self.sqlSablon.sql_sablondetaysil(  
            self.tableWidget_sablondetay.item(self.tableWidget_sablondetay.currentRow(),  
0).text())
```

```
        self.sablondetay()
```

ŞABLON ADI		ŞABLON DETAYLARI							
AÇIKLAMA		HAFTA GÜNÜ	ALET / MAKİNE	AÇIKLAMA	BÖLGE	AĞIRLIK	SET	TEKRAR	ARA
BAŞLANGIÇ DÜZEY EKTOMORF 60-69 Kİ		1	Koşu bandı	15 dk genel ısınma	tüm vücut				
ORTA DÜZEY EKTOMORF 60-69 Kİ		2	dumbell shoulder press	sehpada dik oturur çekilde	omuz	20 kg	5	8	30 sn
İLERİ DÜZEY EKTOMORF 60-69 Kİ		3	biceps curl machine	makinede ön kol çalışma	kol	20 kg	6	8	45 sn
BAŞLANGIÇ DÜZEY EKTOMORF 71 70-79 Kİ		4	bench press machine	makine ile üst göğüs	göğüs	25 kg	4	7	1 dk
ORTA DÜZEY EKTOMORF 70-79 Kİ		5	Koşu bandı	15 dk genel ısınma	tüm vücut				
İLERİ DÜZEY EKTOMORF 70-79 Kİ		6	Tricep pushdown	makine ile ayakta	arka kol	15 kg	7	10	30 sn
BAŞLANGIÇ DÜZEY ENDOMORF 60-69 Kİ		7	Rope Pushdown	makine ile ayakta tam geriye çekilecek	arka kol	15 kg	4	7	45 sn
ORTA DÜZEY ENDOMORF 60-69 Kİ		8	Leg Press machine	makine ile bacak çalışma (%80 itme)	bacak	40 kg	6	10	30 sn
İLERİ DÜZEY ENDOMORF 60-69 Kİ		9	Koşu bandı	15 dk genel ısınma	tüm vücut				
BAŞLANGIÇ DÜZEY ENDOMORF 71 70-79 Kİ		10	dumbell	ayakta kalf kasları çalış	kalf	30 kg	4	12	30 sn
ORTA DÜZEY ENDOMORF 70-79 Kİ		11	Leg Extention machine	makinede üst bacak (tam kaldır)	bacak	40 kg	5	8	1 dk
İLERİ DÜZEY ENDOMORF 70-79 Kİ		12	Sehpa	sehpa üzerinde Hyperextension	sırt		7	8	40 sn

Şekil 3.2.6 Şablon tanımlama ve detaylar

3.2.7 Pilates ve benzeri işlemler

Günümüz fitness merkezleri sadece kişiye özel bir program verip çeşitli alet ve makinelerle bu hareketlerin yapılması olmaktan çıkmıştır. Çoğu fitness merkezleri toplu olarak çeşitli sportif faaliyetleri yapmaktadır. Bunlardan en çok tercih edilen Pilates sporudur. Bu aktivite genelde küçük gruplar halinde belirli saat dilimleri arasında yapılmaktadır. Her üyenin haftada 6 defa geldiği ve üyeye ait çalışma saatleri belirlenmektedir. Pilates modülü ile kişi bu işlemleri yapıp kişiyi takip etmek çok kolay yapılmaktadır. Sabah-öğlen , öğlen akşam olmak üzere 2 sekmeden oluşan bu modülde ilgili zaman dilimine çift tıklanarak kişi listesinden üye seçip haftalık listemizi oluşturabiliriz. Seçilen üye hafta içerisinde 6 gün gelebilmektedir tabloda 6 günden az olan üyelere herhangi bir arka plan rengi oluşmamaktadır. 6 güne eklenen üyeler yeşil 6 günden fazla eklenen üyeler pastel kırmızı renkle gösterilmektedir.

```
def haftaliktablodoldur(self, arama=None):
```

```
    for row in range(self.tableWidget_haftalik_sabah.rowCount()):
```

```
        for col in range(self.tableWidget_haftalik_sabah.columnCount()):
```

```
            if self.tableWidget_haftalik_sabah.item(row, col):
```

```
                if ":" not in self.tableWidget_haftalik_sabah.item(row, col).text():
```

```
                    self.tableWidget_haftalik_sabah.item(row, col).setText("")
```

```
                    self.tableWidget_haftalik_sabah.item(row, col).setBackground(QBrush("white"))
```

```
    for row in range(self.tableWidget_haftalik_oglen.rowCount()):
```

```
        for col in range(self.tableWidget_haftalik_oglen.columnCount()):
```

```
            if self.tableWidget_haftalik_oglen.item(row, col):
```

```
                if ":" not in self.tableWidget_haftalik_oglen.item(row, col).text():
```

```
                    self.tableWidget_haftalik_oglen.item(row, col).setText("")
```

```
                    self.tableWidget_haftalik_oglen.item(row, col).setBackground(QBrush("white"))
```

```
    tablosabah = self.sqlHaftalik.sql_haftaliksabah(arama)
```

```
    tablooglen = self.sqlHaftalik.sql_haftalikoglen(arama)
```

```
    countsabah = [x[2] for x in tablosabah]
```

```
    countoglen = [x[2] for x in tablooglen]
```

```
    counttoplamlam = []
```

```
    for i in countsabah:
```

```
        counttoplamlam.append(i)
```

```
    for i in countoglen:
```

```

    counttoplam.append(i)
for i in tablosabah:
    item = QTableWidgetItem(i[2])
    if counttoplam.count(i[2]) == 6:
        item.setBackground(QBrush("#8aaa96"))
        item.setForeground(QBrush("#ffffff"))
    elif counttoplam.count(i[2]) > 6:
        item.setBackground(QBrush("#da9898"))
        item.setForeground(QBrush("#ffffff"))
    self.tableWidget_haftalik_sabah.setItem(i[0], i[1], item)
for i in tablooglen:
    item = QTableWidgetItem(i[2])
    if counttoplam.count(i[2]) == 6:
        item.setBackground(QBrush("#8aaa96"))
        item.setForeground(QBrush("#ffffff"))
    elif counttoplam.count(i[2]) > 6:
        item.setBackground(QBrush("#da9898"))
        item.setForeground(QBrush("#ffffff"))
    self.tableWidget_haftalik_oglen.setItem(i[0], i[1], item)
self.lineEdit_haftaliksorgula.clear()
self.lineEdit_haftaliksorgula.setFocus()
self.tableWidget_haftalik_sabah.clearSelection()
self.tableWidget_haftalik_oglen.clearSelection()

def haftalikekle(self, table):
    if table.item(table.currentRow(), table.currentColumn()):
        if ":" in table.item(table.currentRow(), table.currentColumn()).text():
            table.clearSelection()
            table.clearFocus()
            return
    elif table.item(table.currentRow(), table.currentColumn()).text() == "":
        kisilistesi = FitnessKisiListesi(
            table.accessibleName(), table.currentRow(), table.currentColumn(), "ekle")
        kisilistesi.exec()

```

```

        kisilistesi.destroy(True, True)
        kisilistesi.deleteLater()
    else:
        kisilistesi = FitnessKisiListesi(
            table.accessibleName(), table.currentRow(), table.currentColumn(), "güncelle")
        kisilistesi.exec()
        kisilistesi.destroy(True, True)
        kisilistesi.deleteLater()
    else:
        kisilistesi = FitnessKisiListesi(
            table.accessibleName(), table.currentRow(), table.currentColumn(), "ekle")
        kisilistesi.exec()
        kisilistesi.destroy(True, True)
        kisilistesi.deleteLater()
    self.haftaliktablodoldur(self.lineEdit_haftaliksorgula.text())

def haftaliksabahsil(self):
    item =
self.tableWidget_haftalik_sabah.item(self.tableWidget_haftalik_sabah.currentRow(),
                                     self.tableWidget_haftalik_sabah.currentColumn())
    if item:
        self.sqlHaftalik.sql_haftaliksil(1, self.tableWidget_haftalik_sabah.currentRow(),
                                         self.tableWidget_haftalik_sabah.currentColumn())
        self.message.setText(item.text() + "\nİptal Edildi")
        self.message.exec()
        self.haftaliktablodoldur(self.lineEdit_haftaliksorgula.text())

    else:
        self.message.setText("Seçili hücrede kayıt yok")
        self.message.exec()

def haftalikoglensil(self):
    item =
self.tableWidget_haftalik_oglen.item(self.tableWidget_haftalik_oglen.currentRow(),

```

```

        self.tableWidget_haftalik_oglen.currentColumn())
if item:
    self.sqlHaftalik.sql_haftaliksil(2, self.tableWidget_haftalik_oglen.currentRow(),
        self.tableWidget_haftalik_oglen.currentColumn())
    self.message.setText(item.text() + "\nİptal Edildi")
    self.message.exec()
    self.haftaliktablodoldur(self.lineEdit_haftaliksorgula.text())
else:
    self.message.setText("Seçili hücrede kayıt yok")
    self.message.exec()

def haftaliktemizle(self):
    self.message.setText("Tablodaki tüm veriler silinsin mi ?\nBu işlem geri alınamaz")
    self.message.withButton()
    ret = self.message.exec()
    if ret == QMessageBox.StandardButton.Yes:
        self.sqlHaftalik.sql_haftaliktumunusil()
        self.haftaliktablodoldur(self.lineEdit_haftaliksorgula.text())
    else:
        pass

```

PİLATES							
SABAH		ÖĞLEN					
PAZARTESİ		SALI	ÇARŞAMBA	PERŞEMBE	CUMA	CUMARTESİ	PAZAR
09:00							
1							
2					MUHAMMED ZEYD POLAT		
3							
4			ATIYE KÖSE POLAT				
5	MUHAMMED ZEYD POLAT	ATIYE KÖSE POLAT					
6				ATIYE KÖSE POLAT			
7							MUHAMMED ZEYD POLAT
8							
10:00							
1							
2		MUHAMMED İYAZ POLAT					
3							
4					ABDURRAHMAN POLAT		
5							
6		ABDURRAHMAN POLAT	MUHAMMED ZEYD POLAT			MUHAMMED İYAZ POLAT	
7							MUHAMMED İYAZ POLAT
8							
11:00							
1							
2							
3							
4							
5							
6	ATIYE KÖSE POLAT	ABDURRAHMAN POLAT			MUHAMMED İYAZ POLAT		ATIYE KÖSE POLAT
7							
8							
12:00							
ARA							

Şekil 3.2.7 Pilates tablosu

3.2.8 Muhasebe gelir gider tablosu

Bu modül karmaşık olan hesap işlemlerini daha kolay, anlaşılabilir ve sorgularla çabuk ulaşılabilir olmasına olanak tanımakla beraber işletmeye ait gelir gider kayıtlarını raporlayabilmektedir (<https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QPieSeries.html>) . Böylelikle kar-zarar hesabı net bir şekilde gözlemlenip gerekli aksiyonlar alınabilmektedir. Üyelere, işletme giderlerine ve kantine ait gelir giderler takip edilebilmektedir.

def muhasebegelirekle(self):

kisilistesi = FitnessKisiListesi(durum="muhasebe")

kisilistesi.exec()

kisi = kisilistesi.kisi

kisilistesi.destroy(True, True)

kisilistesi.deleteLater()

if kisi is not None:

hizmetekle = FitnessHizmetEkle(kisi)

hizmetekle.exec()

hizmetekle.destroy(True, True)

hizmetekle.deleteLater()

self.muhasebegelirtablo()

```
else:  
    return False
```

```
def muhasebegelirsil(self):  
    if self.tableWidget_gelir.selectedItems():  
        idd = self.tableWidget_gelir.item(self.tableWidget_gelir.currentRow(), 0).text()  
        self.sqlMuhasebegelir.sql_muhasebegelirsil(idd)  
        self.muhasebegelirtablo()
```

```
def muhasebegelirtablo(self):  
    self.tableWidget_gelir.clearSelection()  
    self.tableWidget_gelirdetaylar.clearSelection()  
    self.tableWidget_gelirdetaylar.setRowCount(0)  
    veri =  
self.sqlMuhasebegelir.sql_muhasebegelir(self.dateTimeEdit_gelir1.dateTime().toPython(),  
                                         self.dateTimeEdit_gelir2.dateTime().toPython(),  
                                         odemeturu=str(  
                                             self.buttonGroup_muhasebegelir.checkedButton().text()),  
                                         kisi=self.lineEdit_gelirara.text())  
    toplamtutar = sum([x[3] for x in veri])  
    tabloyuolustur(self.tableWidget_gelir, veri)  
    self.label_muhasebe_gelir.setText("{} TL".format(str(toplamtutar)))  
    self.lineEdit_gelirara.setFocus()
```

```
def muhasebegelirdetaytablo(self):  
  
    veri = self.sqlHizmetDetay.sql_hizmetlereklenendetay(  
        self.tableWidget_gelir.item(self.tableWidget_gelir.currentRow(), 1).text())  
    tabloyuolustur(self.tableWidget_gelirdetaylar, veri)
```

```
def muhasebegiderekle(self):  
    gider = FitnessGiderEkle()  
    gider.exec()  
    gider.destroy(True, True)
```

```
gider.deleteLater()
self.muhasibegidertablo()
```

```
def muhasibegidertablo(self):
    odemeler =
self.sqlMuhasebegider.sql_muhasibegider(self.dateTimeEdit_gider1.dateTime().toPython(),
                                         self.dateTimeEdit_gider2.dateTime().toPython())
    tabloyuolustur(self.tableWidget_gider, odemeler)
    self.tableWidget_gelirdetaylar.setContentsMargins(20, 20, 20, 20)
    toplamtutar = sum([x[1] for x in odemeler])
    self.label_muhasibe_gider.setText("{} TL".format(str(toplamtutar)))
    self.lineEdit_gelirara.setFocus()
```

```
def muhasibegidersil(self):
    pass
```

```
def muhasibetabchanged(self, e):
```

```
    if e == 2:
        self.istatistik()
```

```
def sql_muhasibegelir(self, date1=None, date2=None, id_hizmet=None,
odemeturu="TÜMÜ",kisi=""):
```

```
    self.sql_baglan()
```

```
    kisi="% "+kisi+"%"
```

```
    if id_hizmet:
```

```
        query = ("select odemetutari,iskonto from muhasibegelir where id_hizmet=?")
```

```
        param = (id_hizmet,)
```

```
        veri = self.cur.execute(query, param).fetchone()
```

```
    else:
```

```
        if odemeturu == "TÜMÜ":
```

```
            query = ("select * from muhasibegelirdetaylar where eklenmetarihi between ? and ?
and kisi_kantin like ?")
```

```
            param = (date1, date2,kisi)
```

```
        else:
```

```
            query = ("select * from muhasibegelirdetaylar where eklenmetarihi between ? and ?
```



```
and odemesekli=? and kisi_kantin like ?")
    param = (date1, date2, odemeturu,kisi)
    veri = self.cur.execute(query, param).fetchall()
self.sql_kapat()
return veri
```

```
def sql_muhasabegelirekle(self, id_hizmet, odemetutari, iskonto, odemeturu):
    self.sql_baglan()
    query = ("insert into muhasabegelir(id_hizmet,odemetutari,iskonto,odemeturu) values
(?,?,?,?)")
    param = (id_hizmet, odemetutari, iskonto, odemeturu)
    self.cur.execute(query, param)
    self.con.commit()
    self.sql_kapat()
```

```
def sql_muhasabegelirsil(self, idd):
    self.sql_baglan()
    query = ("delete from muhasabegelir where id=?")
    param = (idd,)
    self.cur.execute(query, param)
    self.con.commit()
    self.sql_kapat()
```

FitnesPro

MUHASEBE

GELİR GİDER KASA

HAREKETLER

	AÇIKLAMA	ÖDENEN	ISKONTO	TOPLAM	ÖDEME ŞEKLİ	EKLENME TARİHİ
1	KANTİN	842	0	842	NAKİT	2024-01-29 10:46:32
2	SAFİYE POLAT 5374332267	11000	0	11000	KREDİ KARTI	2024-01-29 20:53:31
3	MUHAMMED ZEVD POLAT 5555566666	12750	0	12750	NAKİT	2024-01-29 20:54:57
4	MUHAMMED İYAZ POLAT 568985656	22000	500	22500	KREDİ KARTI	2024-01-29 20:55:31
5	ABDURRAHMAN POLAT 5337988817	19000	500	19500	NAKİT	2024-01-29 20:55:53

DETAYLAR

	AÇIKLAMA	ÖDEME TÜRÜ	FİYAT
1	FULL PAKET 3 AY (FITNESS + SAUNA + PLATES)	NAKİT	11000
2	SAUNA 3 AY	NAKİT	1750

TOPLAM GELİR: 65592 TL

ARA

28.01.2024 00:00:00

29.01.2024 23:59:59

TÜMÜ NAKİT KREDİ KARTI EFT/HAVALE BORÇ

Şekil 3.2.8.1 Muhasebe gelir

FitnesPro

MUHASEBE

GELİR GİDER KASA

	ÖDEME TUTARI	AÇIKLAMA	ÖDEME TÜRÜ	EKLENME TARİHİ
1	20000	OCAK 2024 KİRA GİDERİ	EFT/HAVALE	2024-01-29 10:00:58
2	15000	ARALIK 2023	NAKİT	2024-01-29 10:46:57

TOPLAM GİDER: 35000 TL

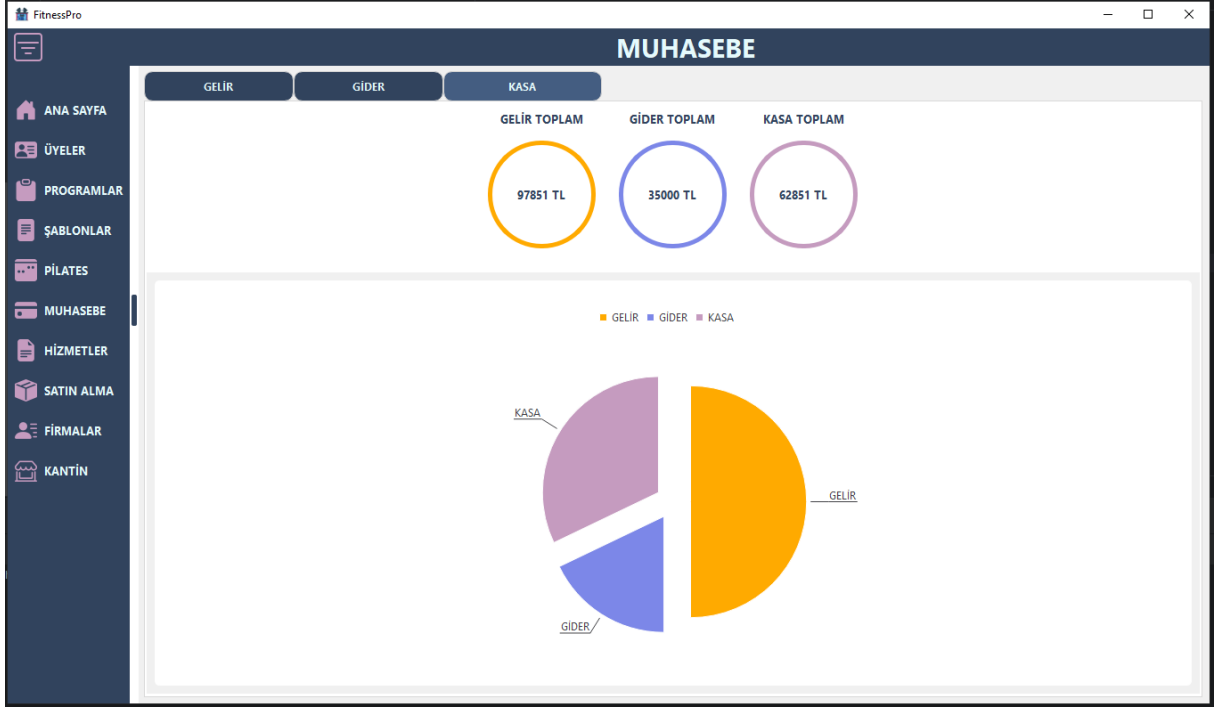
ARA

29.01.2023 00:00:00

29.01.2024 23:59:59

TÜMÜ NAKİT KREDİ KARTI EFT/HAVALE BORÇ

Şekil 3.2.8.2 Muhasebe gider



Şekil 3.2.8.3 Muhasebe istatistik

3.2.9 Hizmetlerin oluşturulması.

Fitness merkezlerinde çeşitli aktiviteler düzenlenmektedir. Bu merkezler sadece vücut geliştirme ile sınırlı kalmayıp pilates, sauna, spa, yüzme ve benzeri faaliyetlerde yürütmektedir. Bu hizmetler önceden fitness uygulamamıza tanımlanıp fiyat belirlendikten sonra üyelere satışı yapılabilmektedir. Hizmetler sekmesinden + butonu ile açılan sayfada oluşturmak istediğimiz hizmete ait bilgileri girdikten sonra kaydetmemiz yetecektir.

FitnesPro

HİZMETLER

HİZMET LİSTESİ

	HİZMET ADI	AÇIKLAMA	FİYAT
1	FITNESS	3 AY	4500
2	PILATES	3 AY	6000
3	FITNESS	6 AY	8500
4	PILATES	6 AY	11000
5	SAUNA	6 AY	3000
6	SAUNA	3 AY	1750
7	FULL PAKET	3 AY (FITNESS + SAUNA + PLATES)	11000

+

-

↻

Şekil 3.2.9.1 Hizmetler listesi

FitnesPro

HİZMETLER

HİZMET LİSTESİ

	HİZMET ADI	AÇIKLAMA	FİYAT
1	FITNESS	3 AY	4500
2	PILATES	3 AY	6000
3	FITNESS	6 AY	8500
4	PILATES	6 AY	
5	SAUNA	6 AY	
6	SAUNA	3 AY	
7	FULL PAKET	3 AY (FITNESS + SAUNA	

HİZMET OLUŞTUR

HİZMET ADI : YÜZME

AÇIKLAMA : 12 SEANS YÜZME DERSİ

FİYAT : 3000

✓

+

-

↻

Şekil 3.2.9.2 Yeni hizmet ekle

3.3 Satınalma , fatura ve stok yönetimi

Fitness merkezleri sadece sportif faaliyetlerinin yürüldüğü bir yer değildir. Çeşitli sporcu gıdaları , içecekler ve buna benzer bir çok şey satabilmektedir. Bu işlemler satınalma modülü ile takip edilebilmektedir. Bu modülde öncelikle sisteme satışı yapılacak olan ürünler tanımlanır bu ürünlerin marka, adı, barkod numaraları, kritik stok seviyeleri gibi işlemler tanımlanır. Daha sonra ürünlere ait faturalar sisteme işlenir böylelikle ürünler stoğa girmiş olur.

```
def urunekle(self):
```

```
    urun = FitnessUrunEkleGuncelle()
    urun.exec()
    urun.destroy(True, True)
    urun.deleteLater()
    self.uruntablosu()
```

```
def urunguncelle(self):
```

```
    urun = FitnessUrunEkleGuncelle(1)
    urunbilgileri = lambda x:
self.tableWidget_urunler.item(self.tableWidget_urunler.currentRow(), x).text()
    urun.urunbilgileri(urunbilgileri(0), urunbilgileri(1), urunbilgileri(2), urunbilgileri(3),
urunbilgileri(4),
                urunbilgileri(5))
    urun.exec()
    urun.destroy(True, True)
    urun.deleteLater()
    self.uruntablosu()
```

```
def urunsil(self):
```

```
    if self.tableWidget_urunler.selectedItems():
        urunid = self.tableWidget_urunler.item(self.tableWidget_urunler.currentRow(), 0).text()
        self.sqlUrun.sqlurunsil(urunid)
        self.uruntablosu()
    else:
        self.message.setWindowTitle("UYARI")
```

```

        self.message.setText("Ürün listesinden bir ürün seçiniz.")
        self.message.exec()

def uruntablosu(self):
    tabloyuolustur(self.tableWidget_urunler,
self.sqlUrun.sql_urunlistesi(self.lineEdit_urunara.text()))
    self.lineEdit_urunara.clear()

def faturatabchanged(self, e):
    if e == 1:
        self.faturatablo()

def faturaekle(self):
    fatura = FitnessFatura()
    fatura.exec()
    fatura.destroy(True, True)
    fatura.deleteLater()
    self.faturatablo()

def faturasil(self):

self.sqlFatura.sql_faturasil(self.tableWidget_fatura.item(self.tableWidget_fatura.currentRow()
, 0).text())
    self.faturatablo()

def faturatablo(self):
    tabloyuolustur(self.tableWidget_fatura,
self.sqlFatura.sql_faturalistesi(self.lineEdit_faturaara.text()))
    self.lineEdit_faturaara.clear()

def faturadetayekle(self):
    faturadetay = FitnessFaturaDetay()
    faturabilgisi = lambda x:
self.tableWidget_fatura.item(self.tableWidget_fatura.currentRow(), x).text()

```

```
faturaid = faturabilgisi(0)
firmaid = faturabilgisi(1)
faturano = faturabilgisi(2)
faturatarihi = faturabilgisi(3)
firmacari = faturabilgisi(8)
firmaresmi = faturabilgisi(9)
faturadetay.faturabilgisi(faturaid, faturano, faturatarihi, firmaid, firmacari, firmaresmi)
faturadetay.exec()
faturadetay.destroy(True, True)
faturadetay.deleteLater()
```

```
def stokguncelle(self):
    if self.tableWidget_urunstokgecmisi.selectedItems():
        print("s")
```

```
def stoklistesi(self):
    urunid = self.tableWidget_urunler.item(self.tableWidget_urunler.currentRow(), 0)
    if urunid:
        tabloyuolustur(self.tableWidget_urunstokgecmisi,
self.sqlFaturaDetay.sql_faturastokgecmisi(urunid.text()))
```


	CARİ AD	RESMİ AD	VERGİ DAİRESİ	E-MAIL	WEB SİTESİ	FATURA NO	FATUR
1	IYZED SPROCU GIDALARI LTD. ŞTİ.	IYZED GIDA TAR.	ANKARA VERGİ	info@iyzedgida.com	www.iyzedgida.com	123123	2024-01
2	IYZED SPROCU GIDALARI LTD. ŞTİ.	IYZED GIDA TAR.	ANKARA VERGİ	info@iyzedgida.com	www.iyzedgida.com	13212ASD	2024-01
3	POLAT GIDA	POLAT GIDE SAN TIC LTD ŞTİ.	GÖKALP	info@polatgida.com	www.polatgida.com	234234234	2024-01

Şekil 3.3.3 İşlenen faturalar

Fatura oluşturmak için öncelikle Şekil 3.3.3 te + butonuna basıp daha sonra şekil 3.3.4 te ilgili alanların doldurulup kaydedilmesi gerekmektedir.

FATURA EKLE

FİRMA: IYZED SPROCU GIDALARI LTD. ŞTİ.

RESMİ AD: IYZED GIDA TAR.

YETKİLİ: MUHAMMED İYAZ POLAT CEP NO: 5551556565

E-MAİL: info@iyzedgida.com TELEFON NO: 4122336565

FATURA TARİHİ: 2024-01-29 FATURA NO:

NOT:

Şekil 3.3.4 Fatura işleme

FitnesPro

SATIN ALMA

FATURA DETAY EKLE

CARİ AD : POLAT GIDA

RESMİ AD : POLAT GIDE SAN TİC LTD ŞTİ.

FATURA ID : 4 FATURA NO : 2024-01-29

FİRMA ID : 3 FATURA NO : 234234234

STOK KARTLARI

MARKA	STOK ADI	
1	ETİ CARAMIO 30 GR	9€
2	BİM HEPİMİZİNO	64
3	WEIDER % 100 WHEY PROTEIN 5000 GR	7€
4	OPTIMUM PROTEIN BAR ÇİLEKLİ 30 GR	5€
5	OPTIMUM PROTEIN BAR KAKAOLU 30 GR	97

MIKTAR : 1 AD. İSKONTO % : 0

BİRİM FİYAT : 0.0 KDV % : 0

TOPLAM : 0

ARA

EKLENENLER

MARKA	STOK ADI	BARKOD NO	MIKTAR	BİRİM	BİRİM FİYAT
1	WEIDER % 100 WHEY PROTEIN 5000 GR	7965465465	10	AD.	1850.0
2	OPTIMUM PROTEIN BAR ÇİLEKLİ 30 GR	564658764	10	AD.	15.0
3	OPTIMUM PROTEIN BAR KAKAOLU 30 GR	97646465654	10	AD.	15.0
4	WEIDER PURE CREATİN 600 GR	654654846546	10	AD.	750.0

TOPLAM TUTAR : 26300.0 TL

KDV : 5260.0 TL

ISKOTO : 2028.0 TL

GENEL TOPLAM : 29532.0 TL

FATURA ARA

Şekil 3.3.5 İşlenen faturanın detaylarını girme

Oluşturulan faturaya çift tıklayarak tanımlı ürün listesinden miktar,adet,birim fiyat,iskonto ve kdv gibi bilgiler girilip + butonu ile listeye eklenir daha sonra fatura kontrol edildikten sonra ilgili sayfaya kapatılır. Şekil 3.3.5

3.4 Firma (Cari kart tanımlama)

Fitness merkezleride diğer işletmeler gibi farklı firmalarla çalışabilmektedir. Örnek verecek olursak kantin için çeşitli gıdaların alınması sporcu yiyecek ve içeceklerinin alınması gibi. Alınan bu ürünlerin stoğa tanımlanması faturalarının işlenmesi ve takibi için firmalar modülüne tanımlama işlemi yapıldıktan sonra Şekil 3.10.4 te fatura işleme modülünde firma – cariad görüntülenir. İlgili firma yada cari seçildikten sonra işlemlere devam edilebilir.

```
def cariekle(self):
```

```
    cari = FitnessCariEkle()
```

```
    cari.exec()
```

```
    cari.destroy(True, True)
```

```
    cari.deleteLater()
```

```
    self.caritablo()
```

```
def carisil(self):
```

```
    if self.tableWidget_carianasayfa.selectedItems():
```

```
        idd = self.tableWidget_carianasayfa.item(self.tableWidget_carianasayfa.currentRow(),  
0).text()
```

```
        cariad =
```

```
self.tableWidget_carianasayfa.item(self.tableWidget_carianasayfa.currentRow(), 1).text()
```

```
        self.message.setText("{}\n\nCari kart silinsin mi ?\n\nBu işlem geri  
alınamaz".format(cariad))
```

```
        self.message.withButton()
```

```
        self.message.exec()
```

```
        if self.message.clickedButton().text() == "Evet":
```

```
            self.sqlCari.sql_carisil(idd)
```

```
            self.caritablo()
```

```
        else:
```

```
            return False
```

```
        else:
```

```
            self.message.setText("Silmek istediğiniz Cari kartı seçiniz")
```

```
            self.message.withOutButton()
```

```
            self.message.exec()
```

```
def caridetay(self):
```

```

caridetay = FitnessCariDetay()
caribilgi = lambda x:
self.tableWidget_carianasayfa.item(self.tableWidget_carianasayfa.currentRow(), x).text()
idd = caribilgi(0)
cariad = caribilgi(1)
resmiad = caribilgi(2)
vergidairesi = caribilgi(3)
verginno = caribilgi(4)
ticaretsdilino = caribilgi(5)
mersisno = caribilgi(6)
notlar = caribilgi(7)
yetkiliad = caribilgi(8)
cepno = caribilgi(9)
telno = caribilgi(10)
email = caribilgi(11)
websitesi = caribilgi(12)
il = caribilgi(13)
ilce = caribilgi(14)
adres = caribilgi(15)

caridetay.caribilgiler(idd, cariad, resmiad, vergidairesi, vergino, ticaretsdilino, mersisno,
notlar, yetkiliad,
                cepno,
                telno, email, websitesi, il, ilce, adres)

caridetay.exec()
caridetay.destroy(True, True)
caridetay.deleteLater()
self.caritablo()

def caritablo(self):
    tabloyuolustur(self.tableWidget_carianasayfa,
self.sqlCari.sql_cariler(self.lineEdit_cariara.text()))
    self.lineEdit_cariara.setFocus()
    self.lineEdit_cariara.clear()

```

FitnesPro

FİRMALAR

	CARİ AD	RESMİ AD	VERGİ DAİRESİ	VERGİ NO	TİCARET SİCİL NO	MERSİS NO	NOTLAR	YETKİ
1	IVZED SPROCU GIDALARI LTD. ŞTİ.	IVZED GIDA TAR.	ANKARA VERGİ	9865656	5656	556	dasd	MUHAMMED İY
2	POLAT GIDA	POLAT GIDE SAN TİC LTD ŞTİ.	GÖKALP	0000	152	566		ABDURRAHMA
3	KIRA GİDERLERİ							

01.01.2000 00:00:00 | 01.01.2000 23:59:59 | Tümü Aktif Pasif

Şekil 3.4.1 Firma/Cari kartlar

FitnesPro

FİRMALAR

	CARİ AD	RESMİ AD	VERGİ DAİRESİ	VERGİ NO	TİCARET SİCİL NO	MERSİS NO	NOTLAR	YETKİ
1	IVZED SPROCU GIDALARI LTD. ŞTİ.	IVZED GIDA TAR.	ANKARA VERGİ	9865656	5656	556	dasd	MUHAMMED İY
2								ABDURRAHMA
3								

CARİ KART EKLE

CARİ AD:

RESMİ AD:

VERGİ DAİRESİ:

VERGİ NO:

TİCARET SİCİL NO:

MERSİS NO:

NOTLAR:

YETKİLİ ADI:

CEP TELEFON:

İŞ TELEFON:

EMİL:

WEB SİTESİ:

İL:

İLÇE:

ADRES:

ARA | 01.01.2000 00:00:00 | 01.01.2000 23:59:59 | Tümü Aktif Pasif

Şekil 3.4.2 Firma / Cari kart ekleme

```

class FitnessCariEkle(QDialog, Ui_CariEkle):
    def __init__(self):
        super().__init__()
        self.setupUi(self)
        self.sqlIlce = SqlFitnessIlce()
        self.sqlCari = SqlCari()
        self.comboBox_il.addItem("")
        for i in self.sqlIlce.sql_il():
            self.comboBox_il.addItem(i[0], i[1])
        self.comboBox_il.currentIndexChanged.connect(self.ilce)
        self.pushButton_kisikaydet.clicked.connect(self.cariEkle)

    def cariekle(self):
        cariad = self.lineEdit_cariad.text()
        resmiad = self.lineEdit_resmiad.text()
        vergidairesi = self.lineEdit_vergidairesi.text()
        vergino = self.lineEdit_vergino.text()
        ticaretsicilno=self.lineEdit_ticaretsicilno.text()
        mersisno=self.lineEdit_mersisno.text()
        yetkiliad = self.lineEdit_yetkiliadi.text()
        cepno = self.lineEdit_cep telefonu.text()
        telno = self.lineEdit_istelefonu.text()
        email = self.lineEdit_email.text()
        websitesi = self.lineEdit_websitesi.text()
        il = self.comboBox_il.currentData()
        ilce = self.comboBox_ilce.currentData()
        adres = self.plainTextEdit_adres.toPlainText()
        notlar = self.plainTextEdit_notlar.toPlainText()
        self.sqlCari.sql_cariekle(cariad, resmiad, vergidairesi, vergino, ticaretsicilno, mersisno,
notlar, yetkiliad, cepno,
            telno, email, websitesi, il, ilce, adres)
        self.close()

    def ilce(self):

```

```

self.comboBox_ilce.clear()
ilceler = self.sqlIIIlce.sql_ilce(self.comboBox_il.currentData())
for i in ilceler:
    self.comboBox_ilce.addItem(i[0], i[1])

```

Şekil 3.4.3 Firma / Cari görüntüleme-güncelleme ve detaylar

```

class FitnessCariDetay(QDialog, Ui_CariDetay):
    def __init__(self, ):
        super().__init__()
        self.setupUi(self)
        self.sqlIIIlce = SqlFitnessIIIlce()
        self.sqlCari = SqlCari()
        self.comboBox_il.addItem("")
        for i in self.sqlIIIlce.sql_ilce():
            self.comboBox_il.addItem(i[0], i[1])
        self.comboBox_il.currentIndexChanged.connect(self.ilce)

```

```
self.pushButton_kisikaydet.clicked.connect(self.guncelle)
```

```
def guncelle(self):
```

```
    cariad = self.lineEdit_cariad.text()
```

```
    resmiad = self.lineEdit_resmiad.text()
```

```
    vergidairesi = self.lineEdit_vergidairesi.text()
```

```
    vergino = self.lineEdit_vergino.text()
```

```
    ticaretsicilno = self.lineEdit_ticaretsicilno.text()
```

```
    mersisno = self.lineEdit_mersisno.text()
```

```
    notlar = self.plainTextEdit_notlar.toPlainText()
```

```
    yetkiliad = self.lineEdit_yetkiliadi.text()
```

```
    cepno = self.lineEdit_ceptelefonu.text()
```

```
    telno = self.lineEdit_istelefonu.text()
```

```
    email = self.lineEdit_email.text()
```

```
    websitesi = self.lineEdit_websitesi.text()
```

```
    il = self.comboBox_il.currentData()
```

```
    ilce = self.comboBox_ilce.currentData()
```

```
    adres = self.plainTextEdit_adres.toPlainText()
```

```
    self.sqlCari.sql_cariguncelle(cariad, resmiad, vergidairesi, vergino, ticaretsicilno,  
mersisno, notlar,
```

```
        yetkiliad,
```

```
        cepno,
```

```
        telno, email, websitesi, il, ilce, adres, self.cariid)
```

```
def ilce(self):
```

```
    self.comboBox_ilce.clear()
```

```
    ilceler = self.sqlIlce.sql_ilce(self.comboBox_il.currentData())
```

```
    for i in ilceler:
```

```
        self.comboBox_ilce.addItem(i[0], i[1])
```

```
def caribilgiler(self, idd, cariad, resmiad, vergidairesi, vergino, ticaretsicilno, mersisno,  
notlar, yetkiliad,
```

```
    cepno,
```

```
    telno, email, websitesi, il, ilce, adres):
```



```
self.cariid = idd
self.lineEdit_cariad.setText(cariad)
self.lineEdit_resmiad.setText(resmiad)
self.lineEdit_vergidairesi.setText(vergidairesi)
self.lineEdit_vergino.setText(vergino)
self.lineEdit_ticaretsicilno.setText(ticaretsicilno)
self.lineEdit_mersisno.setText(mersisno)
self.plainTextEdit_notlar.setPlainText(notlar)
self.lineEdit_yetkiliadi.setText(yetkiliadi)
self.lineEdit_ceptelefonu.setText(cepno)
self.lineEdit_istelefonu.setText(telno)
self.lineEdit_email.setText(email)
self.lineEdit_websitesi.setText(websitesi)
self.comboBox_il.setCurrentText(il)
self.comboBox_ilce.setCurrentText(ilce)
self.plainTextEdit_adres.setPlainText(adres)
```

3.12 Kantin ve ürün satışı

Fitness merkezine ait kantin modülü ile stoktaki ürünlerin satışları ve takibi yapılabilmektedir. Çok satan, kar marjı yüksek olan ürünler tarih aralığına göre raporlanıp böylelikle stoğa alınacak olan ürünlerin seçimi konusunda daha elle tutulabilir bir veri elde edilmiş olur.

```
def kantinstoktablo(self):
    tabloyuolustur(self.tableWidget_kantinurunler,
self.sqlUrun.sql_urunlistesi(self.lineEdit_kantinurunara.text()))

def kantineklenentablo(self):
    tabloyuolustur(self.tableWidget_kantineklenen, self.kantineklenentampon)

def kantinsepttencikar(self):
    item = lambda x:
self.tableWidget_kantineklenen.item(self.tableWidget_kantineklenen.currentRow(), x).text()
self.kantineklenentampon.remove((item(0), item(1), item(2), item(3)))
self.kantineklenentablo()
self.label_kantintoplam.setText("0")
self.label_kantingeneltoplam.setText("0")
self.lineEdit_kantiniskonto.setText("0")
self.kantintoplamtutar()

def kantinsepeteekle(self):
    item = lambda x:
self.tableWidget_kantinurunler.item(self.tableWidget_kantinurunler.currentRow(), x).text()
self.kantineklenentampon.append([item(0), item(1), item(2), item(4)])
self.kantineklenentablo()
self.kantintoplamtutar()

def kantintoplamtutar(self):
    if len(self.lineEdit_kantiniskonto.text()) >= 1:
        toplamtutar = round(sum([float(x[3]) for x in self.kantineklenentampon]), 2)
        iskonto = round(float(self.lineEdit_kantiniskonto.text()), 2)
        geneltoplam = toplamtutar - iskonto
```

```

self.label_kantintoplam.setText(str(toplamtutar))
self.label_kantingeneltoplaml.setText(str(geneltoplaml))
else:
self.lineEdit_kantiniskonto.setText("0")

def kantinsatis(self):
if self.kantineklenentampon.__len__() > 0:
toplamtutar = self.label_kantingeneltoplaml.text()
iskonto = self.lineEdit_kantiniskonto.text()
odemeturu = self.buttonGroup_kantinodemeturu.checkedButton().accessibleName()
id_hizmeteklenen = self.sqlHizmet.sql_hizmetlereklenenekle("K")
for i in self.kantineklenentampon:
i.insert(0, id_hizmeteklenen)
i.pop(2)
i.pop(2)
self.sqlHizmetDetay.sql_hizmetlereklenenendetayekle(self.kantineklenentampon)
self.sqlMuhasebegelir.sql_muhasabegelirekle(id_hizmeteklenen, toplamtutar, iskonto,
odemeturu)
self.kantineklenentampon.clear()
self.kantineklenentablo()
self.message.setText("İşlem Tamamlandı")
self.message.exec()
else:
self.message.setText("Sepet boş")
self.message.exec()

```

```
def sql_hizmetlereklenendetayekle(self, detay):
```

```
    self.sql_baglan()
```

```
    query = (
```

```
        "insert into hizmetlereklenendetay(id_hizmetlereklenen,id_hizmet,fiyat) values (?,?,:)")
```

```
    self.cur.executemany(query, detay)
```

```
    self.con.commit()
```

```
    self.sql_kapat()
```

```
def sql_muhasabegeligirekle(self, id_hizmet, odemetutari, iskonto, odemeturu):
```

```
    self.sql_baglan()
```

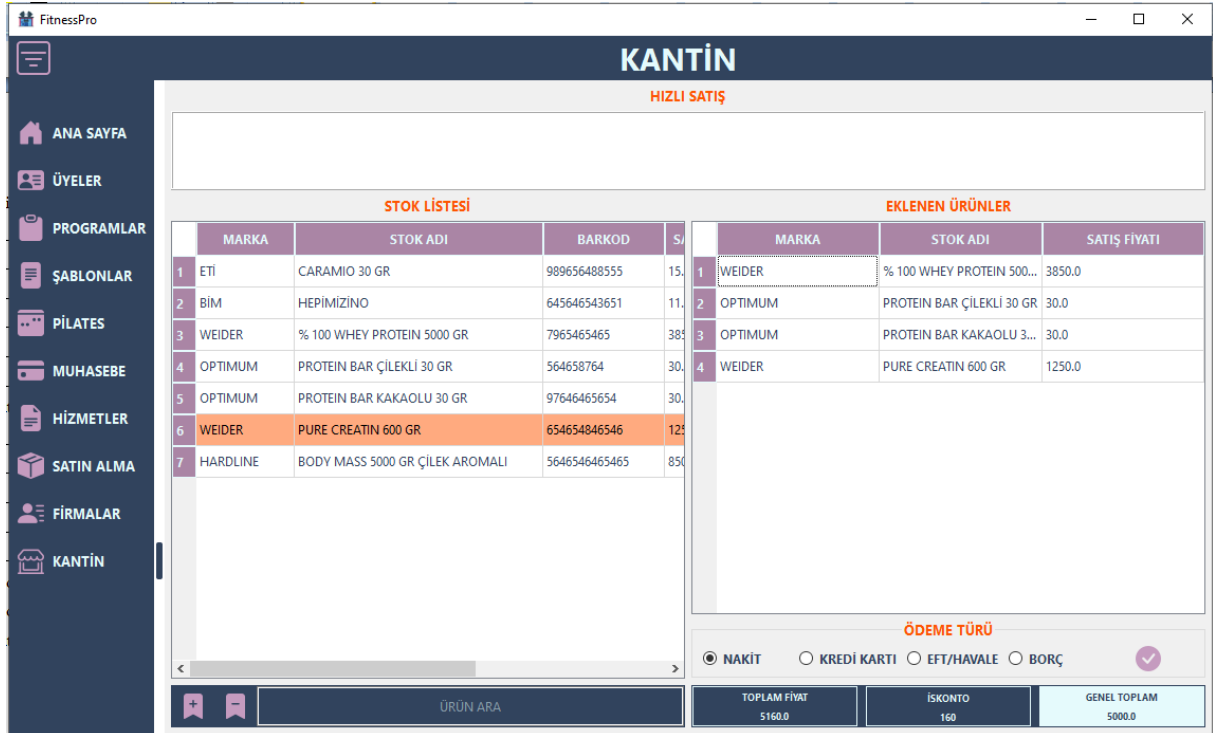
```
    query = ("insert into muhasabegeligir(id_hizmet,odemetutari,iskonto,odemeturu) values  
(?,?,?,?)")
```

```
    param = (id_hizmet, odemetutari, iskonto, odemeturu)
```

```
    self.cur.execute(query, param)
```

```
    self.con.commit()
```

```
    self.sql_kapat()
```



Şekil 3.5 Kantin satış

Kaynaklar

1. <https://doc.qt.io/qtforpython-6/>
2. <https://doc.qt.io/qtforpython-6/PySide6/QtWidgets/QTableWidget.html>
3. <https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QChart.html>
4. <https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QChartView.html>
5. <https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QLineSeries.html>
6. <https://doc.qt.io/qtforpython-6/PySide6/QtCharts/QPieSeries.html>